Application No. 09/933,847
Revised Amendment dated May 8, 2005
Reply to Notice of Non-Compliant Amendment of April 8, 2005

AMMENDMENTS TO THE CLAIMS

This listing of claims will replace all prior versions and listings of claims in the application.

1. (currently amended) ~~An apparatus comprising:~~

~~a mask generator to generate a mask field for an operand having a word length, the mask field defining an operand field within the operand to be operated by an operation, the operand field having a field length; and~~

~~an execution unit coupled to the mask generator to execute the operation on the operand field.~~

A microprocessor system comprising:

a register file of N-bit architectural state registers;

a condition code register evaluated by program flow control instructions or conditional instructions; and

an ALU capable of performing a single operation on an arbitrary bit-field of M bits within two N-bit operands A and B wherein:

M is less than or equal to N.

the bit field of M bits is bounded by end bit Ae and begin bit Ab of the first operand A,

the field width M is equal to Ae - Ab + 1,

the bit field of M bits is right-aligned in the second operand B;

and wherein

condition codes derived from an ALU bit-field operation may be directly saved in the microprocessor's condition code register.

2. (currently amended) ~~The apparatus of claim 1 wherein the mask generator comprises:~~

~~a first decoder to decode a begin position specifier into a begin bit pattern;~~

~~a second decoder to decode an end position specifier into an end bit pattern; and~~

~~a logic circuit coupled to the first and second decoders to combine the begin and end bit patterns into the mask field having the field length limited by the begin and end positions.~~

The microprocessor system of claim 1, in which the condition codes resulting from ALU field operations may be directly used for evaluation of conditional branch instructions, and hence affect the control flow of the program executing on the microprocessor.

3. (currently amended) ~~The apparatus of claim 1 wherein the execution unit comprises:~~

~~a field arithmetic logic unit (ALU) to generate an ALU result using one of an arithmetic and logic operations on the operand field of at least one of first and second ALU operands.~~

The microprocessor system of claim 1 in which the field operation leaves untouched the contextual bits of operand A lying outside the bounds of the field operation, passing them through as bits of the result.

4. (currently amended) ~~The apparatus of claim 3 wherein the execution unit further comprises:~~

~~an operand selector to select a selector operand from a source operand and an immediate operand, the source operand being from a register file.~~

The microprocessor system of claim 1, wherein the condition code register indicates any one or a combination of the conditions:

signed field result overflow within ALU;

unsigned field result overflow within ALU;

ALU operation produced an arithmetic carry-out of the field;

signed field result of ALU is negative;

ALU field result is entirely zero.

5. (currently amended) ~~The apparatus of claim 4 wherein the execution unit further comprises:~~

~~a first barrel shifter coupled to the operand selector to shift the selector operand to generate the first ALU operands according to one of the begin and end positions.~~

The microprocessor system of claim 1, in which the ALU comprises:

a mask generator to delimit the the bit field within the first operand A;

a barrel shifter to shift the operand B Ab bits to the left, effectively multiplying the operand by 2^Ab;

ordinary circuitry for performing arithmetic and logical operations on two operands of width N;

extra circuitry responsive to a mask generator, for the purposes of detecting the condition of the field result; and

a context mux to replace the ALU result bits lying outside the bit-field with the original context bits of operand A.

6. (currently amended) ~~The apparatus of claim 5 wherein the execution unit further comprises:~~

~~a second barrel shifter coupled to the field ALU to shift the ALU result.~~

The microprocessor system system of claim 5, in which the mask generator output is an N-bit mask vector, within which a contiguous set of logic '1' bits or logic '0' bits delimit the bit field of interest, the remainder of bits being logically opposite.

7. (currently amended) ~~The apparatus of claim 5 wherein the execution unit further comprises:~~

~~a context selector coupled to the field ALU to select a field result, on a bit-by-bit basis according to the operand field, from at least one of the first and second ALU operands and the ALU result.~~

The microprocessor system of claim 6, in which the mask generator derives a mask directly from the instruction also specifying the field operation to be performed.

8. (currently amended) ~~The apparatus of claim 6 wherein the execution unit further comprises:~~

~~a context selector coupled to the field ALU to select a field result, on a bit-by-bit basis according to the operand field, from at least one of the first and second ALU operands, the ALU result, and the shifted ALU result.~~

The microprocessor system of claim 6, in which the mask generator derives a mask by decoding bits of the instruction also specifying the field operation to be performed.

9. (currently amended) ~~The apparatus of claim 3 wherein the field ALU comprises:~~

~~N single bit ALUs connected in cascade to generate the field result, the field result including a single bit ALU result.~~

The microprocessor system of claim 6, in which the mask generator derives a mask from extra information saved and associated with the first operand.

10. (currently amended) ~~The apparatus of claim 9 wherein the field result includes at least a condition code representing a condition of the field result.~~

The microprocessor system of claim 6, in which the mask generator derives a mask from extra information saved and associated with the second operand.

11. (currently amended) ~~The apparatus of claim 9 wherein the single bit ALU comprises:~~

~~an adder/subtractor to perform an add/subtraction on the first and second ALU operands and generate a carry output.~~

The microprocessor system of claim 1 in which said field operation is an arithmetic operation including addition or subtraction, after which a result, and optionally, the condition codes for the field, are written back to the architectural machine state.

12. (currently amended) ~~The apparatus of claim 11 wherein the single bit ALU further comprising:~~

~~a zero section to generate a zero condition code using a carry from a less significant section;~~

~~a negative section to generate a out sign bit for the field result using current and next operand fields; and~~

~~an overflow section to generate an overflow bit for the field result using the next operand field.~~

The microprocessor system of claim 1 in which said field operation is an arithmetic compare operation after which only the condition codes for the field are written back to the architectural machine state.

13. (currently amended) ~~The apparatus of claim 2 further comprising:~~

~~a field specifier selector coupled to the mask generator to generate at least one of the begin and end position specifiers.~~

The microprocessor system of claim 1 in which said field operation is a bitwise logical operation including AND, OR, exclusive-OR, or the logical inverses thereof, after which a result, and optionally, the condition codes for the field, are written back to the architectural machine state.

14. (currently amended) ~~The apparatus of claim 13 wherein the field specifier selector generates the at least one of the begin and end position specifiers from at least one of an instruction specifying the operation, a general purpose register, a special purpose register, and a configuration register.~~

The microprocessor system of claim 1 in which said field operation is a field insert operation, in which the first operand is simply passed through the ALU, after which a result, and optionally, the condition codes for the inserted field, are written back to the architectural machine state.

15. (currently amended) ~~The apparatus of claim 1 wherein the operand field is one of a contiguous field and a non-contiguous field.~~

A method, comprising:

performing a single operation on an arbitrary bit-field of M bits within two N-bit operands A and B wherein:

M is less than or equal to N.

the bit field of M bits is bounded by end bit Ae and begin bit Ab of the first operand A,

the field width M is equal to Ae - Ab + 1,

the bit field of M bits is right-aligned in the second operand B; and

optionally saving the result of the operation; and

deriving condition codes from this operation; and

saving said condition codes directly in a condition code register

16. (currently amended) ~~A method comprising:~~

~~generating a mask field for an operand having a word length, the mask field defining an operand field within the operand to be operated by an operation, the operand field having a field length; and~~

~~executing the operation on the operand field, leaving portion outside the operand field unchanged.~~

The method of claim 15, in which the condition codes resulting from operating upon the field are used for evaluation of conditional branch instructions, and hence affect the control flow of the program executing on the microprocessor.

17. (currently amended) ~~The method of claim 16 wherein generating the mask field comprises:~~

~~decoding a begin position specifier into a begin bit pattern; decoding an end position specifier into an end bit pattern; and combining the begin and end bit patterns into the mask field having the field length limited by the begin and end positions.~~

The method of claim 15 in which the contextual bits of operand A lying outside the bounds of the field operation are untouched during the field operation, and are passed through as bits of the results.

18. (currently amended) ~~The method of claim 16 wherein executing the operation comprises:~~

~~generating an ALU result using one of an arithmetic and logic operations on the operand field of at least one of first and second ALU operands.~~

The method of claim 15, wherein generating the condition codes involves one or a combination of:

detecting signed field result overflow within ALU;

detecting unsigned field result overflow within ALU;

detecting the production of an arithmetic carry-out of the field within the ALU

detecting whether the signed field result is negative;

detecting whether the field result is entirely zero.

19. (currently amended) ~~The method of claim 18 wherein executing the operation further comprises:~~

~~selecting a selector operand from a source operand and an immediate operand, the source operand being from a register file.~~

The method of claim 15, in which performing the operation involves:

generating a mask to delimit the the bit field within the first operand A;

shifting the operand B Ab bits to the left, effectively multiplying the operand by 2^Ab;

performing arithmetic and logical operations on two operands of width N;

detecting the condition of the field result; and

replacing the ALU result bits lying outside the bit-field with the original context bits of operand A.

20. (currently amended) ~~The method of claim 19 wherein executing the operation further comprises:~~

~~shifting the selector operand to generate the first ALU operands according to one of the begin and end positions.~~

The method of claim 20, in which generating a mask results in an N-bit mask vector, within which a contiguous set of logic '1' bits or logic '0' bits delimit the bit field of interest, the remainder of bits being logically opposite.

21. (currently amended) ~~The method of claim 20 wherein executing the operation further comprises: shifting the ALU result.~~

The method of claim 21, in which the mask is generated directly from the instruction also specifying the field operation to be performed.

22. (currently amended) ~~The method of claim 20 wherein executing the operation comprises:~~

~~selecting a field result, on a bit-by-bit basis according to the operand field, from the first and second ALU operands and the ALU result.~~

The method of claim 21, in which the mask is generated by decoding bits of the instruction also specifying the field operation to be performed.

23. (currently amended) ~~The method of claim 21 wherein executing the operation comprises:~~

~~selecting a field result, on a bit-by-bit basis according to the operand field, from the first and second ALU operands, the ALU result, and the shifted ALU result.~~

The method of claim 21, in which the mask is generated from extra information saved and associated with the first operand.

24. (currently amended) ~~The method of claim 23 wherein generating the ALU result comprises:~~

~~generating the field result using N single bit ALUs connected in cascade, the field result including a single bit ALU result.~~

The method of claim 21, in which the mask is generated from extra information saved and associated with the second operand.

25. (currently amended) ~~The method of claim 24 wherein the field result includes at least a condition code representing a condition of the field result.~~

The method of claim 15 in which said field operation involves arithmetic, including addition or subtraction, after which a result, and optionally, the condition codes for the field, are written back to the architectural machine state.

26. (currently amended) ~~The method of claim 24 wherein generating the field result comprises:~~

~~performing an add/subtraction on the first and second ALU operands; and~~

~~generating a carry output.~~

The method of claim 15 in which said field operation is an arithmetic compare operation after which only the condition codes for the field are written back to the architectural machine state.

27. (currently amended) ~~The method of claim 26 wherein generating the field result further comprises:~~

~~generating a zero condition code using a carry from a less significant section;~~

~~generating a sign bit for the field result using current and next operand fields; and~~

~~generating an overflow bit for the field result using the next operand field.~~

The method of claim 15 in which said field operation is a bitwise logical operation including AND, OR, exclusive-OR, or the logical inverses thereof, after which a result, and optionally, the condition codes for the field, are written back to the architectural machine state.

28. (currently amended) ~~The method of claim 17 further comprising:~~

~~generating at least one of the begin and end position specifiers using a field specifier selector.~~

The method of claim 15 in which said field operation is a field insert operation, in which the first operand is simply passed through the ALU, after which a result, and optionally, the condition codes for the inserted field, are written back to the architectural machine state.

29. (withdrawn) The method of claim 28 wherein generating at least one of the begin and end position specifiers comprises generating the at least one of the begin and end position specifiers from at least one of an instruction specifying the operation, a general-purpose register, a special-purpose register, and a configuration register.

30. (withdrawn) The method of claim 16 wherein the operand field is one of a contiguous field and a non-contiguous field.

31. (withdrawn) A processor core comprising: a register file having a plurality of registers; a condition code register to store condition codes resulted from an operation; and a field processing unit coupled to the register file and the condition code register to perform an operation, the field processing unit comprising: a mask generator to generate a mask field for an operand having a word length, the mask field defining an operand field within the operand to be operated by the operation, the operand field having a field length, and an execution unit coupled to the mask generator to execute the operation on the operand field, leaving portion outside the operand field unchanged.

32. (withdrawn) The processor core of claim 31 wherein the mask generator comprises: a first decoder to decode a begin position specifier into a begin bit pattern; a second decoder to decode an end position specifier into an end bit pattern; and a logic circuit coupled to the first and second decoders to combine the begin and end bit patterns into the mask field having the field length limited by the begin and end positions.

33. (withdrawn) The processor core of claim 31 wherein the execution unit comprises: a field arithmetic logic unit (ALU) to generate an ALU result using one of an arithmetic and logic operations on the operand field of at least one of first and second ALU operands.

34. (withdrawn) The processor core of claim 33 wherein the execution unit further comprises: an operand selector to select a selector operand from a source operand and an immediate operand, the source operand being from a register file.

35. (withdrawn) The processor core of claim 34 wherein the execution unit further comprises: a first barrel shifter coupled to the operand selector to shift the selector operand to generate the first ALU operands according to one of the begin and end positions.

36. (withdrawn) The processor core of claim 34 wherein the execution unit further comprises: a second barrel shifter coupled to the operand selector to shift the ALU result.

37. (withdrawn) The processor core of claim 35 wherein the execution unit further comprises: a context selector coupled to the field ALU to select a field result, on a bit-by-bit basis according to the operand field, from at least one of the first and second ALU operands and the ALU result.

38. (withdrawn) The processor core of claim 36 wherein the execution unit further comprises: a context selector coupled to the field ALU to select a field result, on a bit-by-bit basis according to the operand field, from at least one of the first and second ALU operands, the ALU result, and the shifted ALU result.

39. (withdrawn) The processor core of claim 36 wherein the field ALU comprises: N single bit ALUs connected in cascade to generate the field result, the field result including a single bit ALU result.

40. (withdrawn) The processor core of claim 39 wherein the field result includes at least a condition code representing a condition of the field result.

41. (withdrawn) The processor core of claim 39 wherein the single bit ALU comprises: an adder/subtractor to perform an add/subtraction on the first and second ALU operands and generate a carry output.

42. (withdrawn) The processor core of claim 41 wherein the single bit ALU further comprising: a zero section to generate a zero condition code using a carry from a less significant section; a negative section to generate a out sign bit for the field result using current and next operand fields; and an overflow section to generate an overflow bit for the field result using the next operand field.

43. (withdrawn) The processor core of claim 32 wherein the field processing unit further comprises: a field specifier selector coupled to the mask generator to generate at least one of the begin and end position specifiers.

44. (withdrawn) The processor core of claim 43 wherein the field specifier selector generates the at least one of the begin and end position specifiers from at least one of an instruction specifying the operation, a general-purpose register, a special-purpose register, and a configuration register.

45. (withdrawn) The processor core of claim 31 wherein the operand field is one of a contiguous field and a non-contiguous field.